

AFRL-IF-RS-TR-2004-346
Final Technical Report
December 2004



EXTRACTING DYNAMIC EVIDENCE NETWORKS

BBN Technologies

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK

STINFO FINAL REPORT

This report has been reviewed by the Air Force Research Laboratory, Information Directorate, Public Affairs Office (IFOIPA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

AFRL-IF-RS-TR-2004-346 has been reviewed and is approved for publication

APPROVED:

/s/
WALTER V. GADZ
Project Engineer

FOR THE DIRECTOR:

/s/
JOSEPH CAMERA, Chief
Information & Intelligence Exploitation Division
Information Directorate

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 074-0188	
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing this collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503				
1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE December 2004	3. REPORT TYPE AND DATES COVERED FINAL Sep 01 – Oct 04	
4. TITLE AND SUBTITLE EXTRACTING DYNAMIC EVIDENCE NETWORKS			5. FUNDING NUMBERS C - F30602-01-C-0204 PE - 62301E PR - EELD TA - 01 WU - 13	
6. AUTHOR(S) Ralph Weischedel				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) BBN Technologies 10 Moulton Street Cambridge MA 02138-1119			8. PERFORMING ORGANIZATION REPORT NUMBER N/A	
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) AFRL/IFED 525 Brooks Road Rome NY 13441-4505			10. SPONSORING / MONITORING AGENCY REPORT NUMBER AFRL-IF-RS-TR-2004-346	
11. SUPPLEMENTARY NOTES AFRL Project Engineer: Walter V. Gadz/IFED/(315) 330-3948 Walter.Gadz@rl.af.mil				
12a. DISTRIBUTION / AVAILABILITY STATEMENT APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.			12b. DISTRIBUTION CODE	
13. ABSTRACT (Maximum 200 Words) BBN's primary goal was to dramatically increase the accuracy of evidence extraction. Using a hybrid of statistical learning algorithms and handcrafted patterns, SERIF achieved 93% of human performance in extracting entities, events, and relations, and 96% of human performance in extracting relations given entities and events. A second performance objective was to be able to extract entities that have names at 80% of human performance. This performance was then further improved in the relation extraction work done in 2004. An additional objective was to have a prototype robust enough that it could extract evidence continually (24x7) from a daily English news feed. All objectives were achieved. BBN's SERIF system also represents a significant advance for extraction systems in architecture and implementation. The combination of general linguistic models trained on preexisting corpora with domain specific components trained for the particular task allows powerful linguistic analysis tools to be brought to bear on extracting the relations and events of a new domain. The use of propositions as an intermediate step was an important part of this strategy, encapsulating the literal meaning of the text from which the target relations could then be derived.				
14. SUBJECT TERMS Evidence extraction, statistical learning algorithms, entities, propositions, ontology, event, relation, predicate, active learning			15. NUMBER OF PAGES 38	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED	18. SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED	19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED	20. LIMITATION OF ABSTRACT UL	

Table of Contents

1	Summary.....	1
1.1	Objectives	1
1.2	Approach and Techniques.....	1
1.3	Accomplishments.....	4
2	Results in 2002 and 2003: Two Simple approaches to learning to extract relations	8
2.1	Relation Finding and Scoring	9
2.2	The Probabilistic Models	11
2.3	Training Data	17
2.4	Results.....	19
3	Results in 2004: Improved Learning algorithm.....	24
3.1	Summary	24
3.2	Core Concept	25
3.3	Model Description	25
3.4	Discussion of Features New in 2004	28
3.5	Performance Maximization.....	29
3.6	Conclusion	30
4	Related work	31
	References.....	31

List of Figures

1. Rapid Example Generation & Training.....	2
2. Illustrating the difference between cascaded models and an integrated model.....	3
3. Explicitly stated links extracted from Russian Organized Crime Corpus.....	5
4. Example of entities and relations automatically extracted from the Russian Organized Crime Corpus.....	6
5. Performance of the three classifiers as a function of training set size.....	21
6. Overall Performance.....	22
7. Learning algorithm outperformed our handcrafted rules on each test set.....	23
8. Improved relation extraction performance in 2004.....	24
9. Relation performance given various feature sets for the 2004 model.....	29
10. Combining models from 2003 and 2004 gives best performance.....	30

List of Tables

1. Type Constraints on Relations.....	10
2. Relation finding performance with a randomly trained model.....	18
3. Comparison of relation finding performance between randomly trained model and model trained with intelligently selected training examples.....	19
4. Relative Performance.....	20
5. Performance given varying training set sizes.....	21

1 SUMMARY

1.1 Objectives

BBN's primary goal was to dramatically increase the accuracy of evidence extraction, in particular, achieving at least 90% of human performance on extracting relations given the entities. A second performance objective was to be able to extract entities that have names at 80% of human performance. An additional objective was to have a prototype robust enough that it could extract evidence continually (24x7) from a daily English news feed.

All objectives were achieved.

1.2 Approach and Techniques

BBN's approach has four main elements:

- Integrate statistical learning algorithms wherever feasible.
- Analyze all text into propositions, i.e., relations among entities, where the relations (predicates) are literally stated in the text.
- Map relations to the most specific relation in the ontology as input for link discovery and pattern learning.
- Integrate general linguistic training (e.g., names and treebanks) and small domain training sets (e.g., relations in terms of the ontology) into the models.

Our extraction engine, SERIF (Statistical Entity & Relation Information Finding), embodies this approach. It uses trained statistical models both for the core linguistic analysis components, like the parser, and for components that depend more on the particular domain, like the model that predicts the semantic type of noun phrases. As shown in Figure 1, the models that do general linguistic analysis can be trained using large preexisting training corpora, while the more domain-specific models like those for the specific target relations are trained on smaller batches of relevant training data annotated for the purpose.

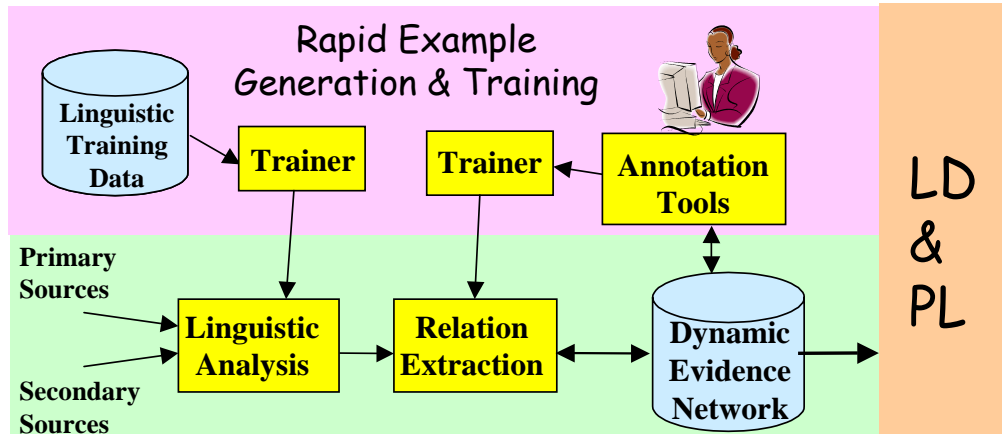


Figure 1: Rapid Example Generation & Training

SERIF’s combined analysis models map the text into its component propositions, which represent the relations among entities that are literally stated in the text. For example, if the text says “Sergey, a Russian envoy in Paris, was shot by unidentified gunmen”, the system would first recognize that “Sergey”, “envoy”, and “gunmen” are all of type Person, and that the first two refer to the same person. The extracted propositions would then include the modification of “envoy” by “Russian”, the “in” connection between the “envoy” and “Paris”, and the “shot” relation between the “gunmen” and “Sergey”. That propositional representation, which represents the literal meaning of the text, is then mapped to the specific target relations, determining here that the envoy was “affiliatedWith” Russia, that the envoy was “inRegion” Paris, and that there had been an AttackOnTangible event involving the envoy as “victim” and the gunmen as “performedBy”.

A new statistical model (Boschee et al., 2003) was developed early in this program for learning to extract relations from the propositional sentence analyses. This algorithm combines two separate approaches that utilize the same training data – one a classifier based on feature vectors extracted from the examples, and one a generative probability model that evaluates the examples expressed as proposition trees. Each approach estimates the probability that a relationship exists between two entity mentions that are syntactically connected. Each bases its estimate on a probability model generated from the distribution of relations across the proposition structures, which combine information from entity extraction and parse trees to represent the basic predicate-argument structure of the sentence. The two scores are then combined. The algorithm is able to generalize accurately across familiar and unfamiliar words and syntactic structures, and

proved able to predict the existence and type of relations roughly as well as carefully handcrafted rules. Further progress was made on this task in 2004, when a new model was developed that was able to expand the set of instances considered to include all pairs of mentions (not just those which are syntactically connected) and to access features of the data previously unavailable to the algorithm due to their complex interdependencies. This new model successfully improved performance beyond the targets already met in 2003.

SERIF also includes the capability to extract relationships from text even when the relationship cannot be resolved as a specific relation in a pre-defined ontology. The proposition can be extracted, retaining the lexical predicate word and the phrase explicitly stating the relation, even if the closest matching predication in the ontology is just that entity X and entity Y are related.

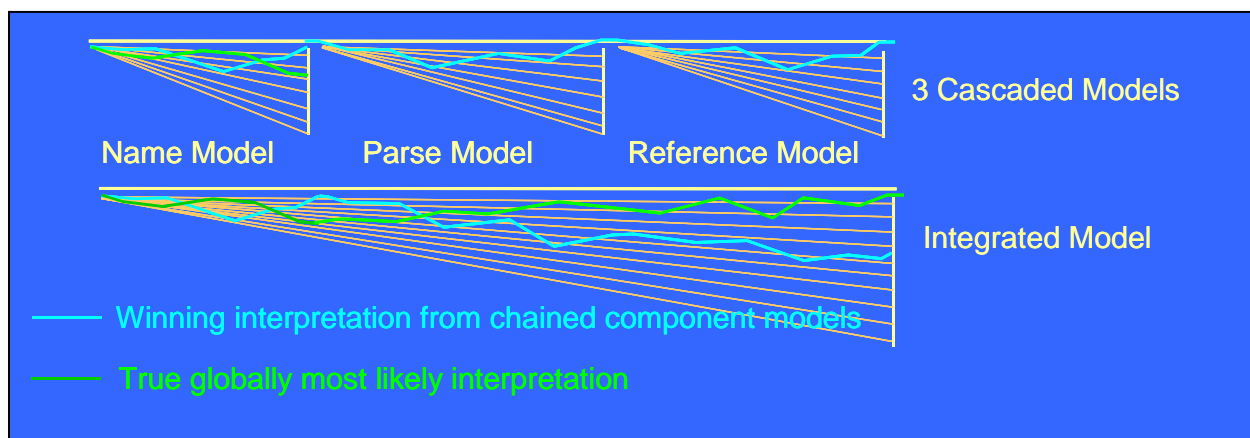


Figure 2: Illustrating the Difference Between Cascaded Models and an Integrated Model

An important advantage in the SERIF design is that it allows for some integration between the model components. Rather than being chained together, with each component outputting only a single best guess, thereby allowing errors to compound themselves throughout the extraction process, SERIF (as shown in Figure 2) is able to maintain a level of integration that allows “later” components to influence earlier decisions. This is achieved by allowing the n-best alternatives from each component, as judged by probability estimates, to be passed to the next component.

SERIF is also easily portable to a new domain and a new set of entity, event, and relation types. Most models were shown to be rapidly adaptable, requiring, at most, new training examples. For models sensitive to changes in the ontology, such as name extraction, no change was required

except for the creation of new annotation data, while for tasks involving general linguistic principles such as co-reference and parsing, no change at all was necessary.

Cross-domain performance was shown to be relatively robust with no changes to either the system or the training data, with only small degradation on the surprise “Al-Qaida” data set. Cross-lingual adaptation is also possible given training examples; SERIF is in the process of being trained for Arabic and Chinese under other efforts.

1.3 Accomplishments

First and foremost, BBN was one of the groups that achieved the program goal of 90% of human performance in evidence extraction. Using a hybrid of statistical learning algorithms and handcrafted patterns, SERIF achieved 93% of human performance in extracting entities, events, and relations and 96% of human performance in extracting relations given entities and events. Note that this surpassed our own initial goals, which were to achieve 80% of human performance on entities with names, and 90% on relations *when given the entities*. This performance was then further improved in the relation extraction work done in 2004.

BBN’s SERIF system also represents a significant advance for extraction systems in architecture and implementation. The combination of general linguistic models trained on preexisting corpora with domain specific components trained for the particular task allows powerful linguistic analysis tools to be efficiently brought to bear on extracting the relations and events of a new domain. The use of propositions as an intermediate step was an important part of this strategy, encapsulating the literal meaning of the text from which the target relations could then be derived.

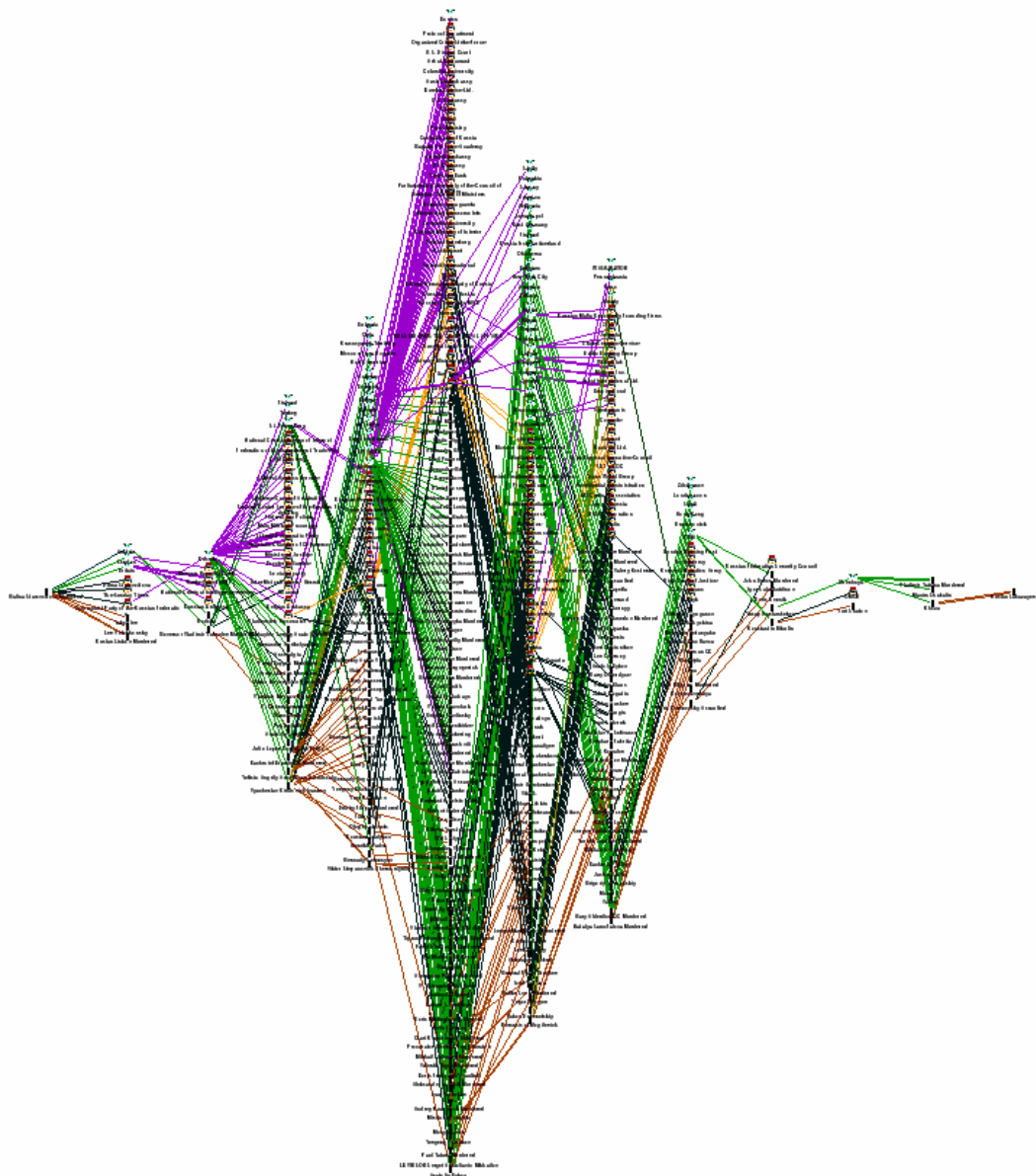


Figure 3: Explicitly stated links extracted from Russian organized crime corpus.

SERIF now serves as the extraction engine at the core of BBN's FactBrowser™ system, which fills a database with information extracted about entities and relations and provides for visualization of that data through a web browser. FactBrowser has been trained and tested on dozens of entity types, and a version of it is now either complete or near completion in both Chinese and Arabic.

Integrating FactBrowser with the VisuaLinksTM link analysis tool has also provided another method of data visualization and analysis, with direct graphical display of the entities found in the corpus and of the relations connecting them. Figure 3 shows the link analysis diagram that resulted when this process was applied to the corpus of documents about Russian organized crime that was provided as part of the EE challenge problem. To produce that graph, BBN's FactBrowser was run over the corpus, producing a database that was then displayed using VisuaLinksTM. The sample shown below in Figure 4 is a small portion of this same extracted network.

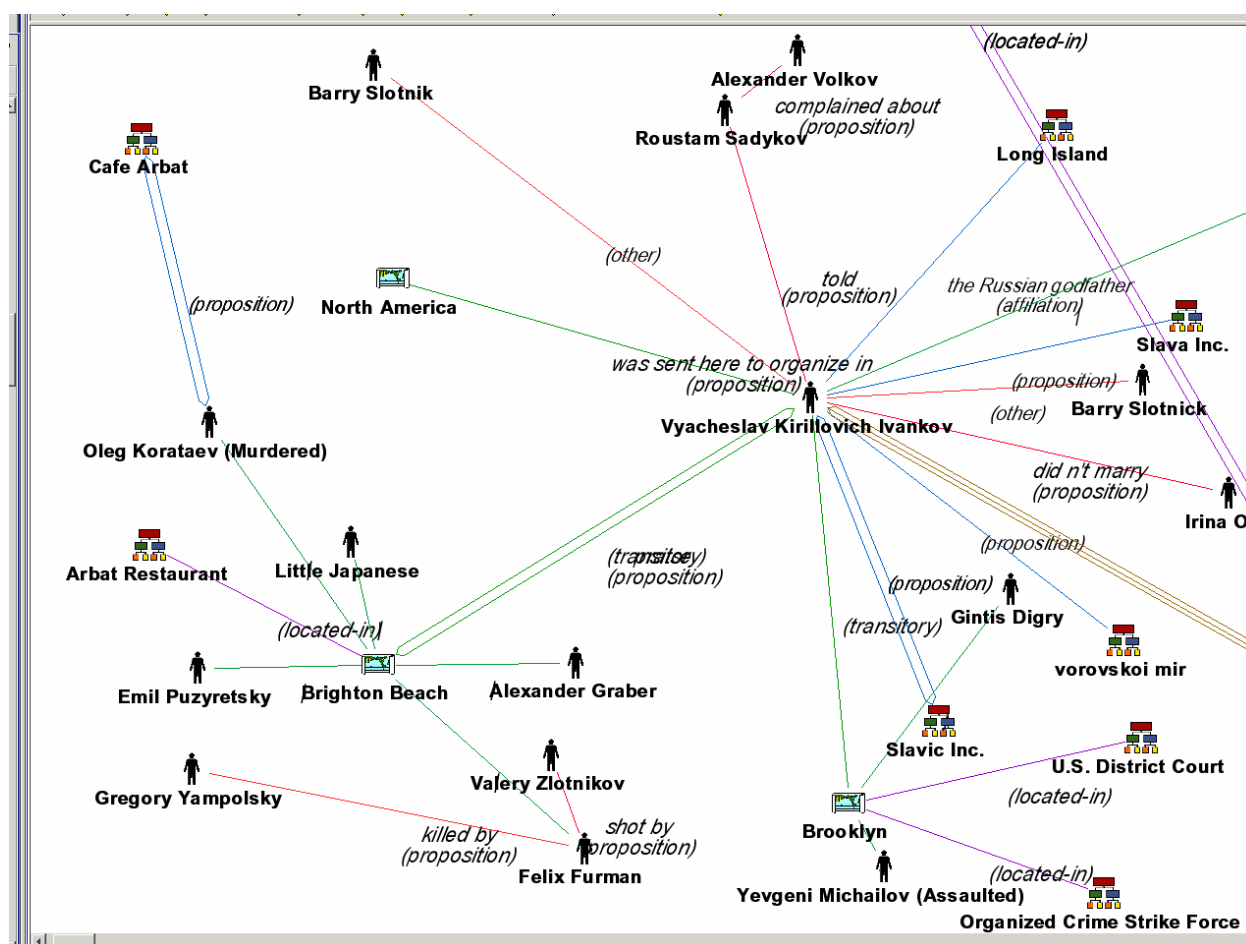


Figure 4: Example of entities and relations automatically extracted from the Russian organized crime corpus

BBN has also delivered SERIF to other projects, including the following:

- The Prototype Platform and the Data Protected Platform, extracting 5 entity types and 24 relation types from Foreign Broadcast information Service (FBIS) sources.

- The TIDES (Translingual Information Detection Extraction and Summarization) Integrated Feasibility Experiment of FY03, where it extracts 5 types of entities and 24 types of relations, and is being ported to Arabic and Chinese.
- BBN's name extraction technology, IdentiFinderTM, is also included in SERIF as a subcomponent. IdentiFinder was independently delivered to Fetch Technologies, and has been distributed as a trained name extraction component for English text to over a dozen sites for research purposes.

2 RESULTS IN 2002 AND 2003: TWO SIMPLE APPROACHES TO LEARNING TO EXTRACT RELATIONS

For more than a decade, the “standard” approach in information extraction from text has been to write handcrafted rules. This standard approach, dating back to the earliest Message Understanding Conferences (MUC), has been applied to a full range of extraction tasks, including names, entities, and relations. In an effort to avoid this necessity for handcrafting, learning techniques are also increasingly being explored for these same tasks. While learning techniques have produced state-of-the-art performance in name tagging (e.g., Bikel, et al., 1999; Palmer, 1999), few attempts (e.g., Miller, et al, 2000; Zelenko, et al., 2002) had been made until recently to learn to extract relations from text, such as news.

The main work we will present was done in the context of the 2002 ACE/EELD/TIDES evaluation, which involves extracting both entities and relations. The Entity Detection and Tracking (EDT) task as defined for that evaluation targets five entity types: person, organization, facility, location, and GPE (geo-political entity, e.g. city, state/province, or country). EDT systems identify and link all textual mentions of those entities, including names, descriptions, and pronouns.

The relation target as defined in the companion Relation Detection and Characterization (RDC) task is inherently more challenging than the Template Relations (TR) task as defined in MUC. In particular, instead of the 3 relation types in MUC TR, there are 24 relation types in RDC (Doddington, 2002; Linguistic Data Consortium, 2002).

In order to extract these relations, we developed three classifiers:

- a tree-based generative model, where we estimate the probability of generating the propositional structures connecting mentions of the two entities,
- a model that represents the connections between entity mentions as a flat vector of features, which is somewhat analogous to traditional handcrafted patterns over chunk parses, and
- a mixture model that combines the above two models.

This latter combination outperformed all but one of the systems in the 2002 Automatic Content Extraction (ACE) evaluation, and performed near human levels as measured by the Linguistic Data Consortium.

We will also report on the performance of the three models as a function of training set size, and on experiments showing the viability of using active learning techniques to maximize the impact of training.

2.1 Relation Finding and Scoring

2.1.1 Task Definition

The RDC task requires detection and characterization of relations between (pairs of) entities.

There are five general types of relations, some of which are further sub-divided, yielding a total of 24 types/subtypes of relations:

- **Role**, the role a person plays in an organization, which can be subcategorized as Management, General-Staff, Member, Owner, Founder, Client, Affiliate-Partner, Citizen-Of, or Other,
- **Part**, i.e., part-whole relationships, subcategorized as Subsidiary, Part-Of, or Other,
- **At**, location relationships, which can be subcategorized Located, Based-In, or Residence,
- **Near**, to identify relative locations and
- **Social**, subcategorizable as Parent, Sibling, Spouse, Grandparent, Other-Relative, Other-Personal, Associate, or Other-Professional.

The type constraints on the arguments to these relations are shown in Table 1.

	FAC	GPE	LOC	ORG	PER
FAC	Part-Of	Located-In	Located-In		
GPE			Located-In Rel-Location	Client	
LOC		Rel- Location			
ORG	Located-In Part-Of	Located-In Based-In Subsidiary	Located-In	Client Subsidiary	Client
PER	Residence Located-In Management General-Staff Owner Founder	Residence Located-In Rel-Location Management General-Staff Affiliate Citizen-Of	Residence Located-In Rel-Location	Management General-Staff Member-Of Owner Founder Affiliate Client	Sibling Spouse Grandparent Other- Relative Other- Personal Associate Other- Professional Member-Of Affiliate Client

Table 1: Type Constraints on Relations. An entry in cell (i,j) indicates a possible relation between entities of type i and those of type j.

2.1.2 Scoring

The scoring in ACE (Doddington, 2002) at both the entity level (EDT) and the relation level (RDC) differs from the recall, precision, and F measures used in earlier information extraction evaluations. Both EDT and RDC scoring assume a two-pass process: first, system output entities (or relations) are mapped against the answer key entities (or relations). Second, a weighted sum of errors is computed from the false alarms (system entities or relations that do not

map to any answer key entity/relation), misses (answer key entities/relations for which there is no mapped system entity/relation), and substitution errors (system entities/relations that do map to an answer key entity/relation, but with some error, e.g., incorrect type). The final score is the “value” of the extracted information, computed by subtracting the weighted sum of errors from 100.

For a system relation to map to an answer key relation, each of the entity arguments in the system output must potentially map to the corresponding entity argument in the answer key, that is, it must share a mention with that corresponding answer key entity. This is much more stringent than the scoring in the MUC measurements of relation extraction; in MUC scoring, a relation could map if any slot was correct, e.g., the type or either entity. Many system relations that would have mapped in MUC to an answer key relation (and thus received substantial partial credit) are not mappable in ACE RDC, resulting in both a missed relation and a false alarm relation.

After an RDC relation has mapped, multiplicative partial credit factors are applied for missed slots, as follows: incorrect time value (90%), incorrect type (60%), incorrect subtype (80%), and incorrectly mapped entity argument (30%). In MUC template relations scoring, each slot of the relation template had equal weight (e.g., entity, relation type, etc.). Partial credit for ACE RDC tends to be less generous than for MUC. For instance, if the system got the type and one entity correct, the partial score in MUC would have been 0.67, but in ACE RDC, it would be 0.30. MUC template relations did not have to predict relation subtype or time, and the cumulative effect of slot errors in RDC multiplies, while in MUC it merely added.

As a result of these differences in task definition and in scoring, while state-of-the-art performance on the MUC TR task of 3 relations was in the low 70s (Marsh & Perzanowski, 1998), state-of-the-art performance in 2002 ACE RDC on 24 relation types was in the low 20s.

2.2 The Probabilistic Models

We approach the task of extracting relations from text by focusing on places in the text where mentions of two different entities are syntactically linked within a single phrase or clause. (Roughly 93% of the target relations are expressed in such a manner.) We train statistical classifiers to judge, based on the entities and the structure that connects them, whether or not a relationship between the entities is being conveyed, and, if so, what kind of relationship it is.

Thus given a pair $\langle M_1, M_2 \rangle$ of noun phrases in text (mentioning a corresponding pair of entities) and the syntactic connection between those two mentions, we use a classifier to predict the pair’s most likely relation type, if any. (We include coreference as an additional identity “relation” type.)

Rather than working directly from the parse trees, the syntactic connections between the entity mentions are first translated into sets of propositions. These propositions attempt to capture the underlying predicate argument structure of the connections without trying to resolve word sense ambiguity. For example, in the phrase “the government was attacked by the party”, the propositional form records that “party” is the logical subject and “government” the logical object of “attack”, without resolving whether what occurred was a speech or a raid.

In this section, we present two separate classification models for finding relations from propositions, one generative and one feature-based, and discuss their relative advantages and disadvantages. Finally, we present a combined model.

2.2.1 A Generative Model using Propositions

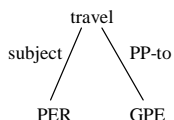
2.2.1.1 Representing Propositions

Propositions represent an approximation of predicate-argument structure and take the form:

predicate (role₁: arg₁, ... , role_n: arg_n)

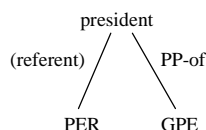
The predicate is typically a verb or noun. Arguments can be either an entity or another proposition. The most common roles include logical subject, logical object, premodifier (for noun predicates), and object of a prepositional phrase modifying the predicate. For example, “Smith went to Spain” is represented as *went(logical subject: Smith, PP-to: Spain)*, and “the U.S. president” is represented as *president(premodifier: U.S.)*.

For our generative model, propositions are represented as trees. Each tree has a left branch (mention M_1) and a right branch (mention M_2). Each node of the tree represents either the predicate of a proposition (non-terminals) or the type of an entity mention (terminals). Each branch has a label representing the role that its child node plays in the parent proposition. So, “Smith traveled to Spain” is represented as:

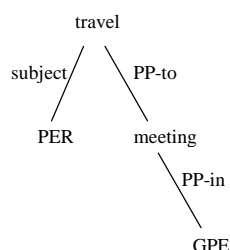


Here, “travel” is the stemmed predicate of the proposition; “subject” and “PP-to” are the roles that the child nodes play; and PER (person) and GPE (geo-political entity) are the entity types of M_1 and M_2 .

Relational nouns are handled analogously. Though M_1 is not syntactically realized as an argument of the predicate, the same graphical representation will suffice. “The president of the U.S.” is represented as:



Where the connection between the two mentions is nested, the tree is extended. For example, “Smith traveled to a meeting in Spain” is represented below. This representation is considerably more flexible than the flat vector model (discussed in 3.2), and it allows us to treat each part of the propositional structure relatively independently.



2.2.1.2 The Generative Probability Model

Our generative model computes the probability that a particular propositional structure conveys a particular relationship by estimating the joint probability of the relation and the structure. This probability is estimated by tracing out the following sequence of decisions that together generate the relation and propositional structure:

- 1) Generate a relation type.
- 2) Given that relation type, decide whether the predicate has only one explicit argument or two. In the first case, only the right branch represents an explicit argument to the predicate; in the second both branches of the tree represent explicit arguments to the predicate.
- 3) Generate the lexical predicate itself.

4) For each branch that represents an argument to a predicate, generate its label: the role that its child node plays in the parent proposition (e.g. logical subject, object, or PP-to).

5) For each child node, generate either an entity type or the predicate of a nested proposition.

When there are nested propositions, their children are generated recursively, using the same probability tables, allowing trees of arbitrary depth.

The dependencies for the probabilistic model of this generative process are as follows:

Element	Depends on
relation type	NONE
top proposition type	relation type
predicate	relation type, top proposition type
branch label	relation type, predicate
node	relation type, branch label, predicate

In addition, the probabilities of branch labels and node types/words are estimated separately for the left and right branches.

Thus the probability of generating “Smith traveled to Spain” as an expression of an At.Located relation would be estimated as the product of the following probabilities:

$P(\text{At.Located})$
 $P(\text{predicate-has-two-args} \mid \text{At.Located})$
 $P(\text{travel} \mid \text{At.Located}, \text{predicate-has-two-args})$
 $P_{\text{left}}(\text{subject} \mid \text{At.Located}, \text{travel})$
 $P_{\text{left}}(\text{PER} \mid \text{At.Located}, \text{subject}, \text{travel})$
 $P_{\text{right}}(\text{PP-to} \mid \text{At.Located}, \text{travel})$
 $P_{\text{right}}(\text{GPE} \mid \text{At.Located}, \text{PP-to}, \text{travel})$

Each of these probability measures is computed as a smoothed mixture of maximum likelihood estimates, using a formula similar to Witten-Bell. The backoff scheme for each component probability is as follows, where “ \rightarrow ” lines show the successive backoff levels for those choices that have backoffs:

$P(\text{relation_type})$

$P(\text{tree_type} \mid \text{relation_type})$
 $P(\text{predicate} \mid \text{relation_type}, \text{tree_type})$
 $\rightarrow P(\text{predicate})$ or $1/10000$ if the predicate is not in vocabulary
 $P_{\text{left/right}}(\text{branch} \mid \text{relation_type}, \text{predicate})$
 $\rightarrow P_{\text{left/right}}(\text{branch} \mid \text{relation_type})$
 $P_{\text{left/right}}(\text{node} \mid \text{relation_type}, \text{branch}, \text{predicate})$
 $\rightarrow P_{\text{left/right}}(\text{node} \mid \text{relation_type}, \text{branch})$
 $\rightarrow P_{\text{left/right}}(\text{node} \mid \text{relation_type})$

2.2.2 The Feature Vector Model

In our second and simpler model, we express each mention pair $\langle M_1, M_2 \rangle$ with its syntactic connection as a set of feature values. Then, given these feature vectors, we use maximum likelihood estimates to select the most likely relation type.

The feature set for two mentions connected by a single proposition is represented as follows:

f_1 = entity type of M_1
 f_2 = entity type of M_2
 f_3 = the syntactic role M_1 plays in the connecting proposition
 f_4 = the syntactic role M_2 plays in the connecting proposition
 f_5 = the (stemmed) predicate of the connecting proposition

Examples:

“IBM hired Smith”
 \rightarrow [ORG PER subject object hire]
 “Smith traveled to Spain”
 \rightarrow [PER GPE subject PP-to travel]
 “the U.S. president”
 \rightarrow [PER GPE referent premod president]

When the connection between the two mentions is a nested structure involving two propositions (e.g. Smith visited a conference in Moscow), f_5 is the stemmed predicate of the first (top) proposition, while f_3 is left to represent the nesting by combining the role that the second proposition plays in the first and the role that the relevant entity mention plays in the second. For example:

“Smith visited a conference in Moscow”
 \rightarrow [PER GPE subject object; PP-in visit]

2.2.2.1 Formal Probability Model

The probability estimate for a relation type is a mixture of two maximum likelihood estimates, one based on all five features, and one based on all but the last feature:

$P_1(\text{relation_type} \mid \langle M_1, M_2 \rangle)$

$$\begin{aligned}
&= P(\text{relation_type}, \{f_{1-5}\}), \text{ and} \\
&P_2(\text{relation_type} \mid \langle M_1, M_2 \rangle) \\
&= P(\text{relation_type}, \{f_{1-4}\})
\end{aligned}$$

Smoothing was done using a formula similar to Witten-Bell.

2.2.2.2 *Analogy to Handcrafted Patterns*

In general, this model’s approach to relation classification is similar to that of traditional handcrafted patterns. The features are not treated independently (unlike in the generative model), and together they define a simple syntactic construction that one might write a pattern to cover. In addition, the model can tolerate significant amounts of noise since it relies on a simplified representation of the situation to make its classification, and one component even ignores the predicate (for cases where insufficient information exists).

For example, this model will likely classify any vector of the form [PER LOC subject PP-in *] as At.Located. This is functionally equivalent to a pattern that says whenever there is a prepositional phrase attached to a verb whose subject is a person, and when that prepositional phrase’s headword is “in” and its object is a location, we find an At.Located relation between the person and the location.

2.2.3 **Advantages and Disadvantages**

The two models have different strengths. The feature vector model performs well on vectors similar to those seen in training, and it also has the ability to generalize across unseen predicate words. For example, as noted above, it can correctly classify “Smith ice-skated in Brazil” as an At.Located relation even though it has never seen the predicate “ice-skate.” On the other hand, it treats each pair of entity types as fundamentally separate from every other pair. Thus, even when the training contains the ice-skating example above—where the feature vector [PER GPE ice-skate subject PP-in] predicts At.Located—the model will still know nothing about the vector [PER FAC ice-skate subject PP-in], e.g., “Smith ice-skated in the local rink”. Finally, because this model ignores the intermediate word in nested constructions, expressions such as “Smith made a deal with Anderson” are unlikely to be correctly classified.

The generative proposition model, on the other hand, can handle nested propositional structures without difficulty. In addition, since the generative model treats the two arguments independently, it can classify relations where it has accurate training information for the probabilities of each of the individual arguments, even if it has not often seen them paired

together (as in the previous example “Smith ice-skated in the local rink”, which the vector model missed).

These independence assumptions in the generative model are often reasonable and help to maximize the impact of the training data. On the other hand, they can sometimes lead the model to overgeneralize. For example, the expression “Chechnya's southern mountains” conveys a Part.Part-Of relation between “Chechnya's southern mountains” and “Chechnya”. The feature vector model is familiar with the exact construction GPE’s LOC, and easily classifies this instance as Part.Part-Of (with probability 0.98). The generative model, on the other hand, analyzes each part of the construction separately and incorrectly classifies it as At.Located, with “Chechnya” being “located” at “Chechnya's southern mountains”. Each aspect of this classification is independently supportable: the model has often seen the first argument of an At.Located relation attached as a possessive premodifier, as in “the man’s hotel room”; it has seen a number of entities (mostly persons) located at mountains; and it has certainly seen many LOCs as the second argument in At.Located relations. Generating GPE as the type of the first argument is not its most likely choice, but it is not so unlikely as to outweigh the other factors. It therefore classifies this instance incorrectly due to the independence assumptions inherent in the model.

2.2.4 Model Combination

The fact that the two models have contrasting advantages and disadvantages suggests trying a combination model. In our combination model, we allow a mention pair to be classified as an RDC relation if and only if both models found some RDC relation present. If the models do both find a relation but disagree as to which relation, we output the one chosen by the feature vector model, under the assumption that the specific relation type classification sometimes subtly depends on the specific combination of the two entity types.

2.3 Training Data

2.3.1 General Approach

We used the following annotation strategy to gather our training data. We ran our current trained entity extraction system over the text, and selected sentences where it found two linked entity mentions. We then presented those individual sentences to our annotators with the two mention spans pre-marked. If the annotator judges the entity marking as correct, they then annotate the

RDC relation, if any, between the two mentions. If the EDT marking is incorrect, the potential relation instance is tagged as incorrect and is not used for training. The full training set is 38K potential relation instances, which include 28K actual relations.

2.3.2 Active Learning

We also ran a set of experiments designed to explore the viability of using active learning techniques to maximize the usefulness of the training data annotated for use by the system. The idea is to begin with a small training set and to add intelligently to it with examples that provide maximal information to the model being trained. For instance, in the case of relation finding, providing the system with a tenth example of “the president of <some organization>” yielding a *Role.Management* relation, is less useful than providing it with the first example of “the <organization> analyst” yielding a *Role.General-Staff* relation. The goal is to develop a process that can focus high-demand training resources on examples that confuse the system, rather than wasting time with examples the system already knows how to classify.

For these experiments, we began with a model trained on 10000 potential relation instances. For the baseline performance, we added examples to the training set at random. Table 2 shows the results of relation-finding with the randomly trained model on the documents and (human-generated) entities from the 2002 evaluation:

training set size	score
10000	40.0
20000	40.2
30000	40.9

Table 2: Relation finding performance with a randomly trained model

For the active learning experiments, we iteratively added examples in increments of 2000. At each iteration, we retrained the model and then added 2000 examples about which the current model was sufficiently ill-informed. Because the relation-finding model is a combination of two separate sub-models, we began by considering the model to be confused about an instance if its sub-models disagreed on the classification. We later expanded the definition to take into consideration the difference in probability between the model’s first-best classification and its

second-best classification. The basic result was as desired: when the model is allowed to intelligently select examples for annotation, its performance improves more quickly than if it is forced to accept examples at random. In fact, whereas the random selection process required 20000 additional examples for the model to reach a score of 40.9, the intelligent process required only 4000, and its performance with 20000 examples well surpassed that of model trained on the randomly selected set:

training set size	score	
	random selection	active learning
10000	40.0	40.0
12000		40.5
14000		41.0
16000		41.3
18000		41.1
20000	40.2	41.5
22000		41.6
24000		41.6
26000		41.9
28000		41.9
30000	40.9	42.3

Table 3: Comparison of relation finding performance between randomly trained model and model trained with intelligently selected training examples

2.4 Results

We were able to measure and compare the performance of the three classifiers on the full RDC task in two ways. First, by using our trained entity extraction technology to find the entities, we could compare the learned classifiers to the state of the art, as reported in the 2002 ACE evaluation. Second, since the Linguistic Data Consortium (Strassel, 2003) has provided their inter-annotator agreement scores on perfect entity markup, we ran our relation classifiers on

perfect EDT input and were able to compare our learning algorithm approach to human performance on this task.

2.4.1 Comparison across Systems

There are five document collections in the September 2002 evaluation test set, from five different sources: newspaper (npaper), newswire (nwire), broadcast news (bnews), and two topic-based EELD sets (eeld1 and eeld2). Results are presented for each source.

Table 4 shows the results of both individual models, the combined model, our former rule-based relation extraction component (using the same trained system to extract entities), and the best reported system from the 2002 RDC evaluation results. The combined model outperforms either of the individual models and our rule-based relation extractor. It comes close to the performance of the best system fielded in that evaluation, which was rule-based.

	Vector model	Generative tree model	Mixture model	Our rule-based	Best RDC system
bnews	14.4	17.2	20.1	21.6	23
eeld_1	20.9	19.8	21.7	20.3	20.2
eeld_2	25.5	20.8	25.5	21.2	17.5
npaper	10.9	11.3	13.6	15.1	16.6
nwire	24.8	24.0	25.2	24.4	29.3
ALL	18.5	18.5	20.6	20.5	unknown

Table 4: Relative Performance: Performance of the vector classifier, proposition classifier, their combination, our rule-based system, and best system are reported on the five test sets and overall.

Table 5 shows the overall performance of each of the three statistical models when trained with different percentages of the available training data; Figure 5 graphs this performance on a logarithmic scale.

	10%	25%	50%	100%
vector	17.0	18.0	18.1	18.5
tree	15.0	17.1	17.0	18.5
mixture	18.1	19.4	19.6	20.6

Table 5: Performance given varying training set sizes.

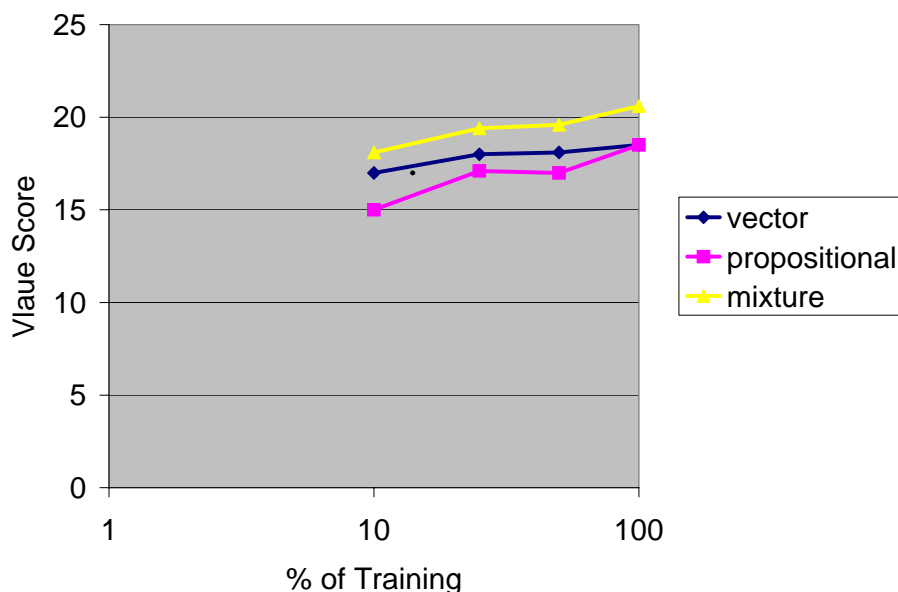


Figure 5: Performance of the three classifiers as a function of Training set size

The vector model in particular seems to be nearly as effective with only 25% of the training data. Presumably, it learns relatively quickly the general sorts of constructions that convey certain relations. The improvement thereafter is likely due largely to specific variations in those constructions based on the lexical information contained in the predicate. Because the space of possible predicates is so vast, it is not surprising the model’s improvement levels off.

The generative proposition model and combined models, on the other hand, are more complex and benefit more from the expansion of the training set; note that performance has not leveled off, suggesting that the generative proposition model is under-trained. This is further supported by the success of the active learning experiments reported on in section 2.3.2.

2.4.2 Comparison with Human Performance

Lastly, we compare the system's performance with human performance. Thanks to (Strassel, 2003), there is an inter-annotator estimate based on sampling the work of annotators experienced at the 2002 RDC task and starting from the same entity markup. The scorer, when applied to the differing annotators' 2002 RDC markup, gives a value of 32.

Though not on the same documents, the closest comparison of our algorithm achievable on this task involves running the learning algorithm on the official evaluation data, and supplying it with the correct entity markup (rather than with the errorful output of our EDT system, as was done in the previous section). Testing in this style, too, as shown in Figure 6, the learned model slightly outperforms our former rule-based relation finder. Figure 7, which breaks down the comparison between the learned model and our former rule-based system into more detail, shows that the learning algorithm outperforms our state-of-the-art handcrafted rules on each of the test sets described in the previous section.

Finally, comparing the learned model's score in Figure 6 to the inter-annotator agreement score cited above indicates that the learning algorithm also performs quite respectably compared to human inter-annotator agreement; however, that comparison, while indicative, is not rigorous on two grounds:

The test sets are not identical

The human performance is a raw inter-annotator agreement score based on correct entity annotation; scores of a single annotator against an adjudicated answer key may be higher.

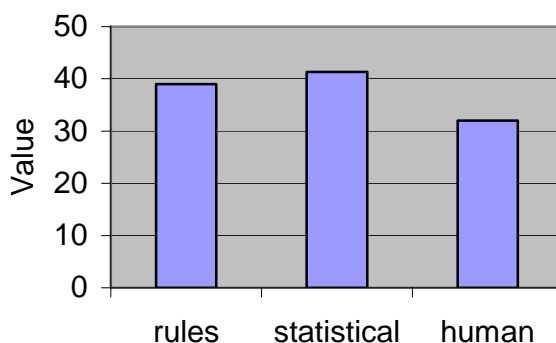


Figure 6: Overall performance.

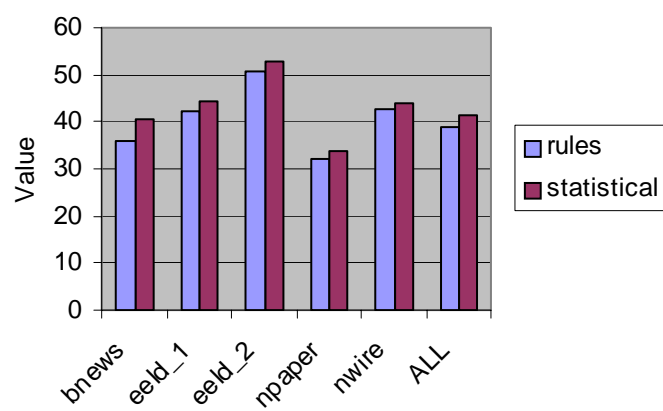


Figure 7: The learning algorithm outperformed our handcrafted rules on each test set.

3 RESULTS IN 2004: IMPROVED LEARNING ALGORITHM

3.1 Summary

Our primary accomplishment in 2004 was improved performance in extracting relations from text using trained statistical systems. We developed a new system designed to capitalize on the information not accessible to our 2003 system. This 2004 system outperformed the approach to relation finding described in section 2 and provides a new, easily modifiable component that can be retrained for any target relation set.

We measured overall improvement by comparing performance on the 2004 ACE/TIDES task¹. This task is similar in form to the 2002 ACE RDC task described in section 2.1.1, with seven relation types and 24 relation subtypes. We trained both systems on the same set of annotated data and tested both on the 2004 ACE/TIDES evaluation corpus. The result was a clear gain in performance as evident in Figure 8.

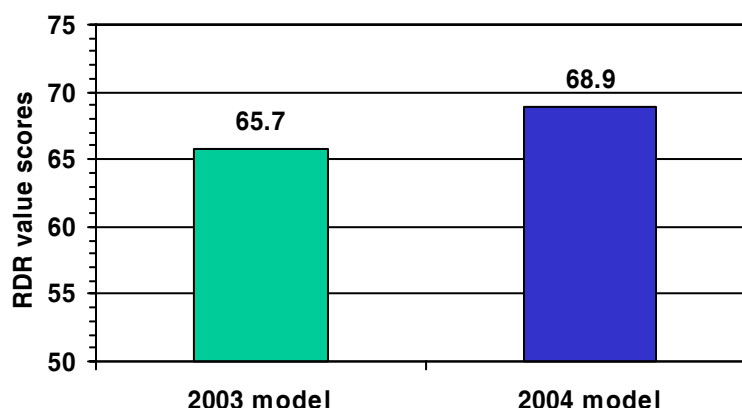


Figure 8: Improved relation extraction performance in 2004.

These results represent the performance of each relation classification system on the “connect-the-dots” task, which measures the ability of the system to detect and classify relationships given entities (as input). This is the clearest way to evaluate RDR performance in isolation from other potentially confusing factors such as co-reference of mentions of entities. Scores are calculated using the 2004 ACE scorer with default weights.

¹ The 2004 ACE evaluation plan, available at <http://www.nist.gov/speech/tests/ace/ace04/doc/ace04-evalplan-v7.pdf>, contains a full description of the task and of the scoring metric used in the evaluation.

3.2 Core Concept

The core concept driving the development of the 2004 system was the desire to explore a wider and more diverse feature space. The 2003 system, as previously described, was a probabilistic model that considered pairs of mentions linked by predicate-argument structures derived from parse trees. That 2003 system used a limited set of features based on that predicate-argument structure, and all such features were required to be independent of each other or have their dependencies specifically managed by the model’s carefully constructed back-off schemes.

The driving motivation behind the development of a new relation finder was a desire to move beyond that restricted set of features to a variety of inter-dependent features. For example, we believe that there is significant and useful information gained by extracting semantic structures from large unannotated corpora. This kind of information, however, seems best represented as a collection of potentially overlapping descriptions of a word or construction: for instance, one might want to say that the word “manager” is often found in the same contexts as “coordinator” and simultaneously say that the word “manager” is occasionally found in the same contexts as the word “vendor.” These two assertions (drawn from real, automatically extracted word clusters) are, however, not independent of each other. The more such non-independent features introduced into a model, the more difficult it becomes to design a generative model that adequately manages all such dependencies.

One central decision made, therefore, was to begin experimenting with discriminative learning algorithms—which have no such feature independence constraints—in order to take advantage of the wide set of features potentially useable in such a framework. The algorithm that drives the relation classifier here is a perceptron-style Viterbi training algorithm.

Development of the new system was then able to expand into this larger feature space. This primarily included the use of word clusters extracted from unannotated corpora, WordNet synsets, surface-level features connecting pairs of mentions, and features derived from predicate-argument structures. Results of these explorations are described in detail below.

3.3 Model Description

3.3.1 Algorithm

The system considers each pair of mentions in a sentence independently. Possible classifications for that pair include all target relation types as well as a classification indicating no relation is

present (“NONE”). For each possible classification of a mention pair, a set of features is extracted. For example, when considering whether “Bush did not visit Moscow” conveys a *Physical.Located* relation, a feature might be expressed as:

Physical.Located negated-proposition,

(corresponding to the physical, located relation being expressed in a negated proposition.)

Each such feature has had a weight assigned to it in training, roughly corresponding to how likely it is that such a feature exists in a correctly classified instance. So, in this example, one would expect the value of this feature weight to be low (or negative), since it is unlikely that a *Physical.Located* relation is expressed by a negated proposition. On the other hand, one would expect the value of *NONE negated-proposition* to be fairly high, since most negated propositions do not convey relations.

Given these features and their weights, then, the model makes its decision by selecting whichever classification generates the features whose summed weight is the highest. This is a basic perceptron-style model. After the relation classification is finalized, argument ordering (e.g., whether A hired B or vice versa) is determined by a simple maximum likelihood model based on the entity types and the predicate-argument connection (if any).

In training, the feature weights are determined by iterating over a set of training instances, decoding them one at a time and adjusting the weights whenever the system makes a mistake. Specifically, if the system misclassifies a training instance, it decrements by some fixed amount the weights of the features extracted from the incorrect classification of the mention pair, and increments (by the same fixed amount) the weights of the features that would have been extracted given the correct classification. In all the experiments discussed here, the system iterates over the full training set five times. Finally, the final feature weights are taken to be the average weight for each feature over the course of the training.

3.3.2 Feature Templates

The relation classification model deployed in the 2004 ACE/TIDES evaluation used 11 feature templates. Descriptions are given below, as well as the specific feature(s) that would be extracted for the following simple example:

“*The assailant fled Moscow the day after the murder*”: *Physical.Located* relation

The two mentions of entities are “the assailant” and “Moscow.”

- 1) The entity types of the two mentions involved
Physical.Located PER GPE
- 2) The entity and mention types of the two mentions involved
Physical.Located PER Nominal GPE Name
- 3) The predicate-argument structure connecting the two entities, if any, using a stemmed version of the predicate
Physical.Located “flee” <subject> <object>
- 4) The predicate-argument structure connecting the two entities, if any, using a stemmed version of the predicate, plus the entity types of the mentions involved
Physical.Located “flee” <subject> PER <object> GPE
- 5) If the proposition is negative, a feature indicating such
none;
- 6) The simple predicate-argument structure connecting the two entities, if any, replacing the predicate with WordNet synsets (of varying size) that include the predicate
Physical.Located “run, turn tail, escape” <sub> <obj>
Physical.Located “leave, go forth, go away” <sub> <obj>
- 7) The simple predicate-argument structure connecting the two entities, if any, replacing the predicate with word clusters (of varying size) that include the predicate
Physical.Located cluster-32 <sub> <obj>
Physical.Located cluster-243 <sub> <obj>
Physical.Located cluster-3057 <sub> <obj>
- 8) The token(s) between the two mentions, plus the entity types of the mentions
Physical.Located PER fled GPE
- 9) The parts of speech of the token(s) between the two mentions, plus the entity types of the mentions
Physical.Located PER VBD GPE
- 10) If one mention modifies another (as in “the Russian president”), the modified word along with the type of modification and the entity type of the modifier—so, for instance,
Employee.Management president <premod> GPE
- 11) If each mention is the object of a prepositional phrase modifying the same noun phrase, a feature indicating such. This gives a handle on the relationship between mentions that might be parsed (correctly or incorrectly) like [the hotel [in *Moscow*] [near *the river*]].

The word clusters described in template (6) are n-gram word clusters generated as described in (Brown et al. 1992), derived from a corpus of 100 million words of Reuters news data.

3.4 Discussion of Features New in 2004

3.4.1 Semantic features

The first new type of feature we wanted to explore is covered by feature templates (6) and (7). These templates generate features that encode semantic information extracted from outside resources—either from WordNet or from word clusters generated from unannotated corpora. The assumption is that although the predicate “flee” may never have been seen in our annotation, it appears in the same word cluster or WordNet synset as “leave,” which we may have seen. Knowing that the two words are connected gives us an additional handle on such an instance.

Obviously each predicate word is likely a member of many WordNet synsets, ranging from the most specific (“president”) to the most general (“person”, “living thing”). The structure of the word clusters is analogous: the clustering is done by organizing the N most frequent words in a corpus into a tree based on the similarity of the contexts they appear in, and a cluster is just a branch of that tree. Each branch higher in the tree includes a set of lower (more “specific”) branches, just as a higher synset in the WordNet tree includes all of its hyponym sets.

Including a feature for every branch a word appears in (or every synset that includes the word) will generate a great deal of redundancy. On the other hand, picking only one cluster or synset for each word obviously restricts the usefulness of the information, dictating a specificity or generality level that may not be appropriate for any given case. We experimented with different ways of selecting features using these hierarchical semantic groupings. In the end, we settled on using every third WordNet synset in the hierarchy and not allowing the most general synsets (e.g. “entity”). For word clusters we used a similar scheme, selecting three preset levels of specificity and using clusters of those sizes (and only those sizes) in our features.

3.4.2 Surface-level features

The second type of new feature is expressed by templates (8) and (9). These templates explore surface-level structure that may contain information missed by a model that relies fully on propositional structure for its information. For instance, in the phrase “his wife and daughter,” the parser will rarely create a structure where “his” modifies “daughter.” The 2003 system would for this reason usually miss the relation between “his” and “daughter.” On the other hand, the surface-level feature “*PER wife and PER*” is quite indicative (as well as common enough to

occur in the training data), so that with the addition of such a feature the 2004 system correctly classifies this instance as SOC.Family.

3.4.3 Results of feature space expansion

Figure 9 below shows the impact of adding both sets of additional features to the model, compared both to the 2003 model and to a baseline 2004 model that uses only the features that were available to the 2003 model (proposition-based, no word clusters or WordNet). The third column shows the improvement gained from adding the “semantic” features generated by templates (6) and (7). The fourth column shows the impact of adding the surface-level (“string”) features generated by templates (8) and (9).

It should be noted that without the advantage of new features, the 2004 model does not match the performance of the old model. This is not surprising; given the same features, a carefully constructed generative model should out-perform a relatively more simple discriminative algorithm.

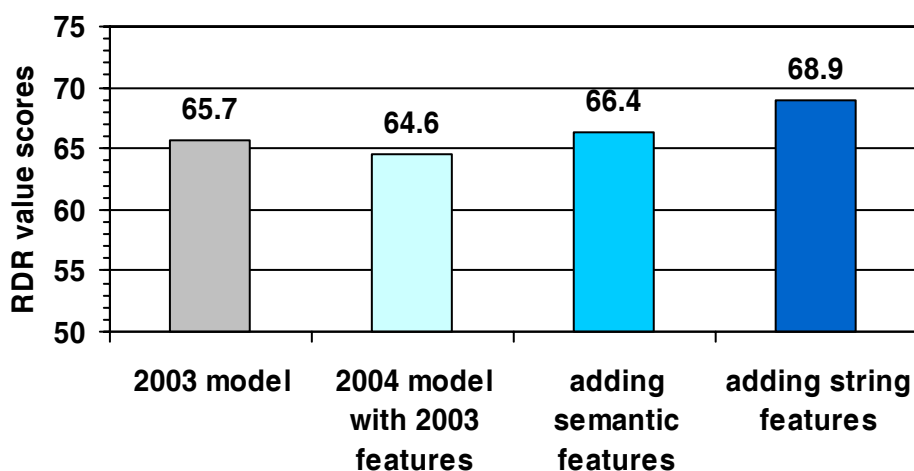


Figure 9: Relation performance given various feature sets for the 2004 model.

3.5 Performance Maximization

The new statistical model outperformed the 2003 model but still under-predicted relations. To address this problem, we took advantage of the fact that the new model is significantly different in structure from the 2003 model. This difference means that, although on the whole, the 2004 model performs better than the 2003 model, there are instances where the 2003 model can correctly predict a relation and the 2004 model does not. Because of this and specifically

because the 2003 model has very high precision, we were able to use the old model to improve the recall of the new: if the new model did not predict a relation between two mentions, but the old model did, we allowed the old model’s prediction to be added to the full 2004 output. This operation, analogous to the previous mixing of the two 2003 sub-models, successfully improved performance by more than 2 points. Figure 10 below shows the 2003 baseline, the discriminative model using the 2004 features, and the full 2004 system that includes input from the old model.

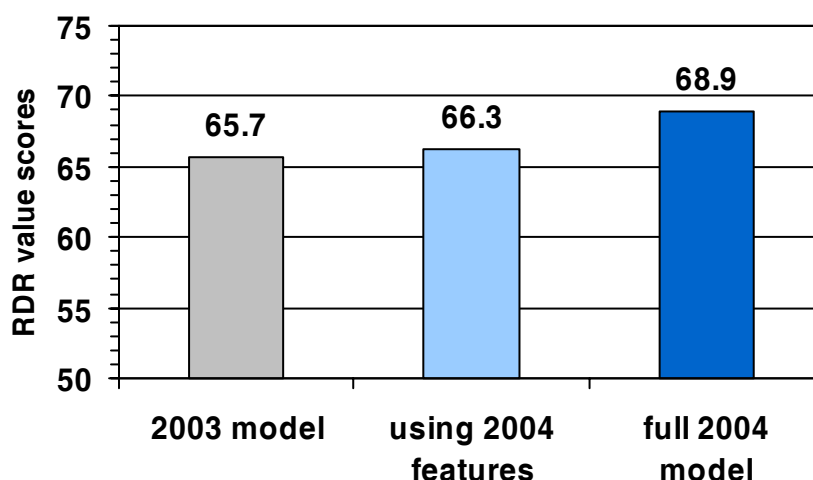


Figure 10: Combining models from 2003 and 2004 gives best performance.

3.6 Conclusion

The 2004 system successfully expanded the feature space to allow a wider and more diverse set of features, and to take advantage of information unavailable to the 2003 system. It was shown to improve performance over the 2003 system, specifically by the addition of such information. This system was submitted as a part of the 2004 ACE/TIDES evaluation, where it was the top-performing system participating in the connect-the-dots RDR evaluation (extract relations, given entities). Its performance was state-of-the-art, among the top performing systems, in the composite RDR task of detecting entities and relations among those detected entities.

4 RELATED WORK

Given a set of sentences annotated with relations, Miller, et al. (2000) describes a procedure that rewrites automatically generated parse trees for those sentences into relation-augmented trees; those trees form the training data for a statistical parser. A simple traversal of the tree converts the relation-augmented tree produced by the parser into the extracted relations. This system was formally evaluated in MUC-7 on the TR relations (person works for organization, organization located at place, and organization makes product).

Zelenko, et al. (2002) apply support vector machines (SVM) to extract relations based on shallow parsing. The SVM was applied to ACE RDC after that publication with excellent results.

Like Miller, our approach uses full parsing, but the classifier is based on propositions rather than parse trees.

Like Zelenko, our approach views the task as a classifier applied to text mentioning entities, though we explored mixtures of several classifiers. Unfortunately, both efforts are promising and warrant further research and development.

Our use of word cluster features builds on work of Miller et al. (2004), which used word cluster features for name finding. That approach combines the distributional word clustering of Brown et al. (1990) with discriminative modeling methods trained using the voted perceptron approach described in Collins and Duffy (2002).

References

- Bikel, D., Schwartz, R., and Weischedel, R. "An algorithm that learns what's in a Name," *Machine Learning* 34, pp. 211-231, (1999).
- Boschee, E., Miller, S., Weischedel, R. "Rapid Annotation through Human-Machine Collaboration." HLT-2002.
- Brown, P., Della Pietra, V., deSouza, P., Lai, J., Mercer, R. "Class-based n-gram models of natural language." *Computational Linguistics*, 1990.
- Chinchor, N. *Overview of MUC-7*. http://www.itl.nist.gov/iad/894.02/related_projects/muc/proceedings/muc_7_toc.html (1998).

- Collins, M. Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms, EMNLP, 2002
- Collins, M., and Duffy, N. New Ranking Algorithms for Parsing and Tagging: Kernels over Discrete Structures, and the Voted Perceptron, ACL, 2002
- Doddington, G. “The ACE 2002 Evaluation Plan”, <http://jaguar.ncsl.nist.gov/ace/doc/ACE-EvalPlan-2002-v06.pdf>, (2002).
- Hasegawa, T., Sekine, S., and Grishman, R. Discovering Relations among Named Entities from Large Corpora, Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL04), pp. 415-422, 2004.
- Marsh, E. & Perzanowski, D. *MUC-7 Evaluation of IE Technology: Overview of Results*. http://www.itl.nist.gov/iad/894.02/related_projects/muc/proceedings/muc_7_toc.html. (1998).
- Miller, S., Guinness, J., and Zamanian, A. Name Tagging with Word Clusters and Discriminative Training, Proceedings of HLT/NAACL 2004, pp. 337-342 (2004).
- Miller, S., Ramshaw, L., Fox, H., and Weischedel, R. “A Novel Use of Statistical Parsing to Extract Information from Text”, In *Proceedings of 1st Meeting of the North American Chapter of the ACL*, Seattle, WA, pp.226-233, (2000).
- Lee, L. and Pereira, F. “Distributional similarity models: Clustering vs. nearest neighbors.” *Proceedings of the 37th ACL*, pp. 33-40 (1999).
- Linguistic Data Consortium. “ACE Phase 2: Information for LDC Annotators”, <http://www ldc.upenn.edu/Projects/ACE2/>, (2002).
- Palmer, D., Burger, J., Ostendorf, M. “Information Extraction from Broadcast News Speech Data,” *Proceedings Of The DARPA Broadcast News Workshop, February 28-March 3*, Morgan Kaufmann Publishers, pp 41-46 (1999).
- Pereira, F., Tishby, N., and Lee L. “Distributional Clustering of English Words. *Proceedings of the 31st ACL*, pp. 183-190 (1993).
- Roark, B., Saraclar, M., Collins, M., and Johnson, M. Discriminative Language Modeling with Conditional Random Fields and the Perceptron Algorithm, Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL04), pp. 47-54, 2004.

Strassel, Stephanie, personal communication, 2003.

Taskar, B., Abbeel, P., and Koller, D. Discriminative Probabilistic Models for Relational Data, Proceedings of the Eighteenth Conference on Uncertainty in AI (UAI02), 2002.

Taskar, B., Wong, M.F., Abbeel, P., and Koller, D. Link Prediction in Relational Data, Neural Information Processing Systems Conference (NIPS03), 2003.

Zelenko, D., Aone, C., and Richardella, A. “Kernel Methods for Relation Extraction.” In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, Philadelphia, PA, pp. 71-78, (2002).